Course:ARTIFICIALINTELLIGENC ETITLE: MARKET BASKET INSIGHTS
PHASE 3 SUBMISSION

TEAM MEMBERS:

MOHANA PRIYA.M

(au311521106061)

MOUNICKA G S

(au31152110602)

PADMA PRIYA .M

(au311521106069)

YAMINI.H

(au 311521106114)

INTRODUCTION:

In today's data-driven world, extracting valuable insights from vast datasets is a fundamental aspect of making informed decisions in various domains, such as retail, marketing, healthcare, and more. Advanced association analysis techniques and visualization tools play a pivotal role in uncovering hidden patterns, relationships, and trends within data, ultimately leading to enhanced insights and more effective strategies.

Data Collection: Obtain the dataset that contains transaction data. This data typically includes information about items purchased together in different transactions.

1. **Data Cleaning**:
   - Remove any duplicates or irrelevant records.
   - Handle missing data, if any, by either removing rows with missing values or imputing them.
   - Ensure data consistency and integrity.
2. **Data Transformation**:
   - Convert the data into a suitable format, often a transaction list or a binary matrix where rows represent transactions, and columns represent items. Each cell is marked as 1 if the item is in the transaction and 0 otherwise.
3. **Support Threshold**: Determine a minimum support threshold, which is the minimum percentage of transactions an itemset (combination of items) must appear in to be considered frequent. This threshold helps in reducing the number of itemsets to be analyzed.
4. **Apriori Algorithm** (or other association rule mining algorithms):
   - Apply the Apriori algorithm or a similar algorithm to find frequent itemsets. The algorithm iteratively discovers itemsets that meet the minimum support threshold.
   - Generate association rules from frequent itemsets.
5. **Rule Pruning**: Optionally, you can prune rules based on additional criteria like confidence, lift, or interest to focus on the most meaningful associations.
6. **Loading the Processed Data**: Depending on your analysis tool, load the preprocessed data, frequent itemsets, and association rules into a data structure that can be used for insights and further analysis.
7. **Analysis and Insights**: Analyze the generated association rules to gain insights into item relationships. This can include identifying frequently co-purchased items, cross-selling opportunities, and other market basket insights.

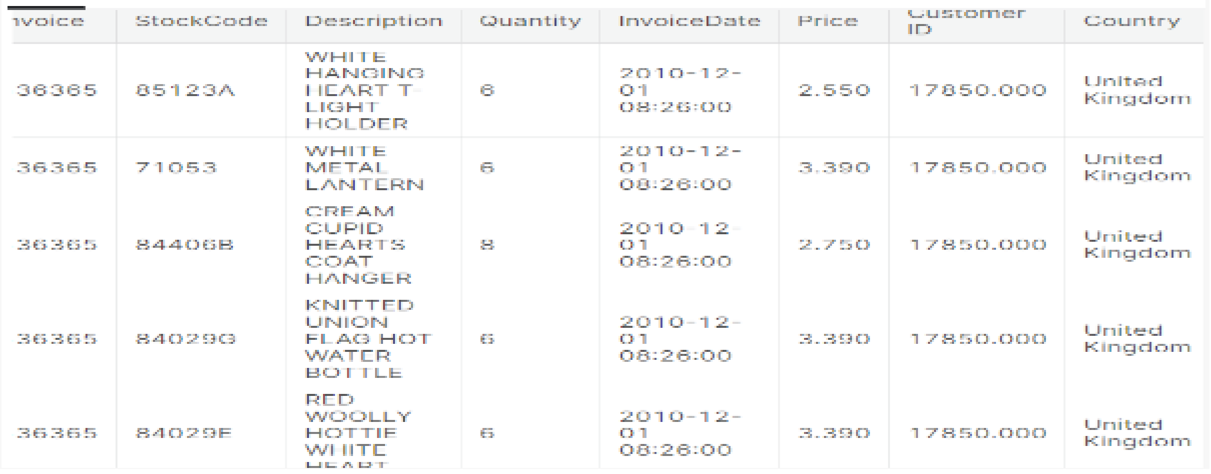
The exact tools and libraries you use for these tasks may vary depending on your preference and the size of your dataset. Common choices include Python with libraries like pandas, scikit-learn, and mlxtend for data preprocessing and association rule mining. For larger datasets, you might want to use distributed data processing frameworks like Apache Spark.

Remember that market basket analysis is a valuable technique for understanding customer behavior and making data-driven business decisions, such as optimizing product placements, running targeted promotions, and improving inventory management.

As a first step, I load all the modules that will be used in this notebook

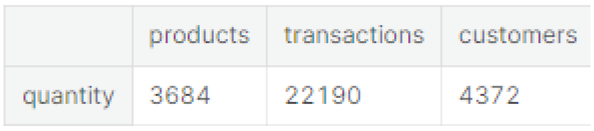
```
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
import seaborn as sns
import datetime, nltk, warnings
import matplotlib.cm as cm
import itertools
from pathlib import Path
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.metrics import silhouette_samples, silhouette_score
from sklearn import preprocessing, model_selection, metrics, feature_selection
from sklearn.model_selection import GridSearchCV, learning_curve
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix
from sklearn import neighbors, linear_model, svm, tree, ensemble
from wordcloud import WordCloud, STOPWORDS
from sklearn.ensemble import AdaBoostClassifier
from sklearn.decomposition import PCA
from IPython.display import display, HTML
import plotly.graph_objs as go
from plotly.offline import init_notebook_mode,iplot
init_notebook_mode(connected=True)
warnings.filterwarnings("ignore")
plt.rcParams["patch.force_edgecolor"] = True
plt.style.use('fivethirtyeight')
mpl.rc('patch', edgecolor = 'dimgray', linewidth=1)
%matplotlib inline
```

Then, I load the data



ıvoice	StockCode	Description	Quantity	InvoiceDate	Price	Customer ID	Country
36365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.550	17850.000	United Kingdom
36365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.390	17850.000	United Kingdom
36365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.750	17850.000	United Kingdom
36365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.390	17850.000	United Kingdom
36365	84029E	RED WOOLLY HOTTIE WHITE HEART	6	2010-12-01 08:26:00	3.390	17850.000	United Kingdom

Customers and products



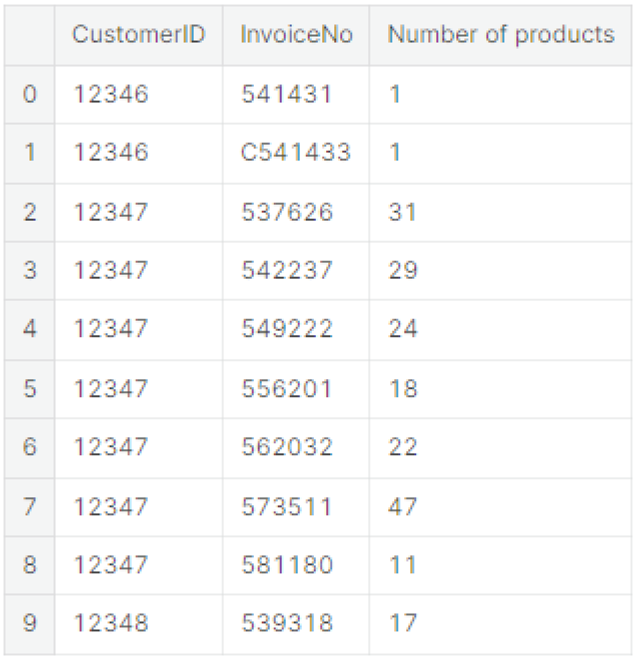
	products	transactions	customers
quantity	3684	22190	4372

It can be seen that the data concern 4372 users and that they bought 3684 different products.
The total number of transactions carried out is of the order of ∼∼22'000.
Now I will determine the number of products purchased in every transaction:

INPUT:

```
temp = df_initial.groupby(by=['CustomerID', 'InvoiceNo'], as_index=False)['Inv
oiceDate'].count()
nb_products_per_basket = temp.rename(columns = {'InvoiceDate':'Number of produ
cts'})
nb_products_per_basket[:10].sort_values('CustomerID')
```

OUTPUT:



	CustomerID	InvoiceNo	Number of products
0	12346	541431	1
1	12346	C541433	1
2	12347	537626	31
3	12347	542237	29
4	12347	549222	24
5	12347	556201	18
6	12347	562032	22
7	12347	573511	47
8	12347	581180	11
9	12348	539318	17

Data encoding

The dataframe `transactions_per_user` contains a summary of all the commands that were made. Each entry in this dataframe corresponds to a particular client. I use this information to characterize the different types of customers and only keep a subset of variables:

INPUT :

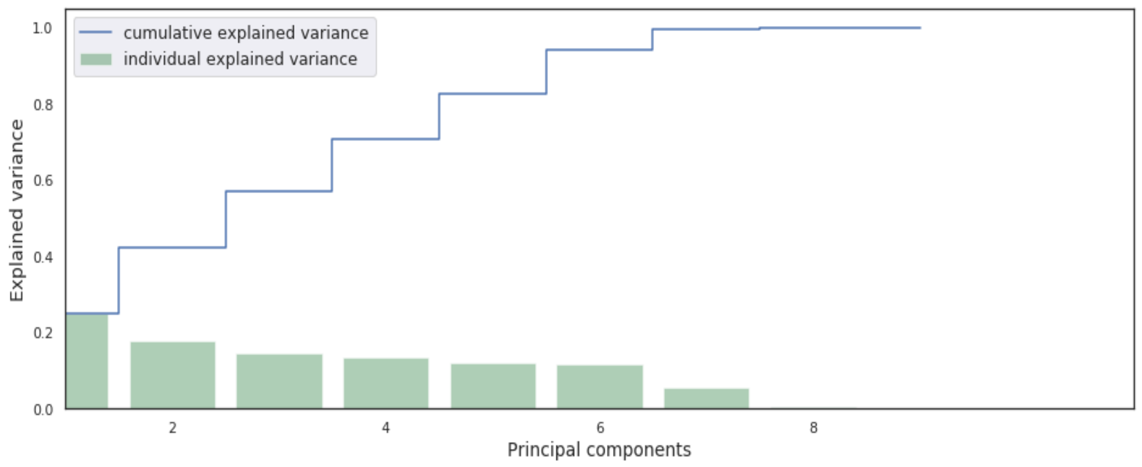
```
list_cols = ['count','min','max','mean','categ_0','categ_1','categ_2','categ_3
','categ_4']
#_____________________________________________________________
selected_customers = transactions_per_user.copy(deep = True)
matrix = selected_customers[list_cols].as_matrix()
```

```
scaler = StandardScaler()
scaler.fit(matrix)
print('variables mean values: \n' + 90*'-' + '\n' , scaler.mean_)
scaled_matrix = scaler.transform(matrix)
pca = PCA()
pca.fit(scaled_matrix)
pca_samples = pca.transform(scaled_matrix)

fig, ax = plt.subplots(figsize=(14, 5))
sns.set(font_scale=1)
plt.step(range(matrix.shape[1]), pca.explained_variance_ratio_.cumsum(), where='mi
d',
         label='cumulative explained variance')
sns.barplot(np.arange(1,matrix.shape[1]+1), pca.explained_variance_ratio_, alpha=0
.5, color = 'g',
            label='individual explained variance')
plt.xlim(0, 10)

ax.set_xticklabels([s if int(s.get_text())%2 == 0 else '' for s in ax.get_xticklab
els()])

plt.ylabel('Explained variance', fontsize = 14)
plt.xlabel('Principal components', fontsize = 14)
plt.legend(loc='best', fontsize = 13);
fig, ax = plt.subplots(figsize=(14, 5))
sns.set(font_scale=1)
```


cumulative explained variance
individual explained variance
Explained variance
Principal components